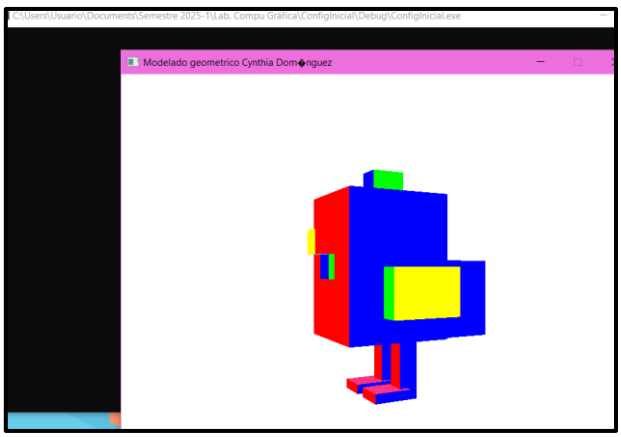
Domínguez Reyes Cynthia Berenice
314148632

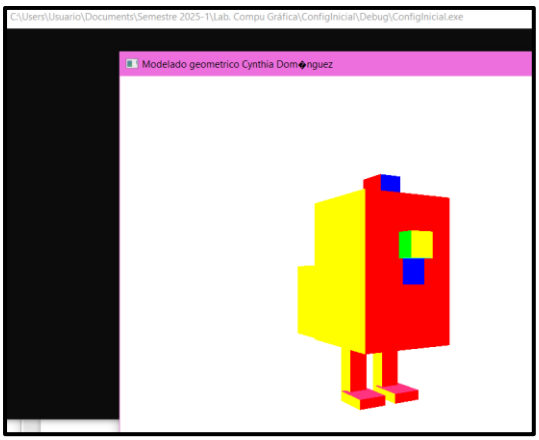
MODELADO

Desarrollo:

Para esta práctica trabajamos con el modelado de cubos para simular una figura box art. Para desarrollar la actividad elegí tomar de referencia la siguiente imagen:

Al replicarla con código obtuve de resultado la siguiente figura:





Ajustes realizados:

Modifique los valores del incremento en las teclas del teclado para manipular la figura ya que, en mi equipo (con los valores que tenía) se movía muy rápido.

```
void Inputs(GLFWwindow *window) {
    if (glfwGetKey(window, GLFW_KEY_ESCAPE) == GLFW_PRESS)  //GLFW_RELEASE
        glfwSetWindowShouldClose(window, true);
    if (glfwGetKey(window, GLFW_KEY_D) == GLFW_PRESS)
        movX += 0.03f;
    if (glfwGetKey(window, GLFW_KEY_A) == GLFW_PRESS)
        movX -= 0.03f;
    if (glfwGetKey(window, GLFW_KEY_PAGE_UP) == GLFW_PRESS)
        movY += 0.03f;
    if (glfwGetKey(window, GLFW_KEY_PAGE_DOWN) == GLFW_PRESS)
        movY -= 0.03f;
    if (glfwGetKey(window, GLFW_KEY_W) == GLFW_PRESS)
        movZ -= 0.03f;
    if (glfwGetKey(window, GLFW_KEY_S) == GLFW_PRESS)
        movZ += 0.03f;
    if (glfwGetKey(window, GLFW_KEY_RIGHT) == GLFW_PRESS)
        rot += 0.4f;
    if (glfwGetKey(window, GLFW_KEY_LEFT) == GLFW_PRESS)
        rot -= 0.4f;
}
```

Para poder dibujar los cubos es necesario establecer las líneas de código para la vista de la ventana donde veremos nuestra figura.

```
view = glm::translate(view, glm::vec3(movX,movY, movZ));
model = glm::scale(model, glm::vec3(3.0f, 1.0f, 2.0f));
view = glm::rotate(view, glm::radians(rot), glm::vec3(0.0f, 1.0f, 0.0f));
GLint modelLoc = glGetUniformLocation(ourShader.Program, "model");
GLint viewLoc = glGetUniformLocation(ourShader.Program, "view");
GLint projecLoc = glGetUniformLocation(ourShader.Program, "projection");


glUniformMatrix4fv(projecLoc, 1, GL_FALSE, glm::value_ptr(projection));
glUniformMatrix4fv(viewLoc, 1, GL_FALSE, glm::value_ptr(view));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));


glBindVertexArray(VAO);
```

Posteriormente tomé un cubo como base para el cuerpo de la figura.

```
//Cubo base
model = glm::mat4(1.0f);
model = glm::scale(model, glm::vec3(1.0f, 1.5f, 1.1f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);
```

A partir de ese cubo, comencé a dibujar los cubos que fueron dando forma a la figura, dándole diferentes escalas y moviéndolos de posición. También agregué rotaciones para poder ver los colores de las otras caras de los cubos.

```
//Figuraa


//pico
model = glm::mat4(1.0f);
model = glm::scale(model, glm::vec3(0.25f,0.25f, 0.5f));
model = glm::translate(model, glm::vec3(0.0f, 1.0f, 1.0f));
model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

model = glm::mat4(1.0f);
model = glm::scale(model, glm::vec3(0.25f, 0.25f, 0.35f));
model = glm::translate(model, glm::vec3(0.0f, 0.0f, 1.25f));
model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//Cresta
model = glm::mat4(1.0f);
model = glm::scale(model, glm::vec3(0.25f, 0.25f, 0.35f));
model = glm::translate(model, glm::vec3(0.2f, 3.5f, -0.1f));
model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);
```

```
//Ala
model = glm::mat4(1.0f);
model = glm::scale(model, glm::vec3(0.2f, 0.5f, 0.75f));
model = glm::translate(model, glm::vec3(3.0f, -0.5f, -0.25f));
model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//Cola
model = glm::mat4(1.0f);
model = glm::scale(model, glm::vec3(1.0f, -0.8f, 0.75f));
model = glm::translate(model, glm::vec3(0.0f, 0.42f, -0.89f));
//model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//Pata der
model = glm::mat4(1.0f); //inicializamos la matriz
model = glm::scale(model, glm::vec3(0.2f, 0.6f, 0.2f)); //Tamaño de la pata
model = glm::translate(model, glm::vec3(1.0f, -1.8f, -1.0f)); //Posicion de la pata
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//patita
model = glm::mat4(1.0f); //inicializamos la matriz
model = glm::scale(model, glm::vec3(0.3f, 0.1f, 0.4f)); //Tamaño de la pata
model = glm::translate(model, glm::vec3(0.7f, -13.1f, 0.03f)); //Posicion de la pata
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//Pata izq
model = glm::mat4(1.0f); //inicializamos la matriz
model = glm::scale(model, glm::vec3(0.2f, 0.6f, 0.2f)); //Tamaño de la pata
model = glm::translate(model, glm::vec3(-1.0f, -1.8f, -1.0f)); //Posicion de la pata
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//patita
model = glm::mat4(1.0f); //inicializamos la matriz
model = glm::scale(model, glm::vec3(0.3f, 0.1f, 0.4f)); //Tamaño de la pata
model = glm::translate(model, glm::vec3(-0.7f, -13.1f, 0.03f)); //Posicion de la pata
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

glBindVertexArray(0);
```